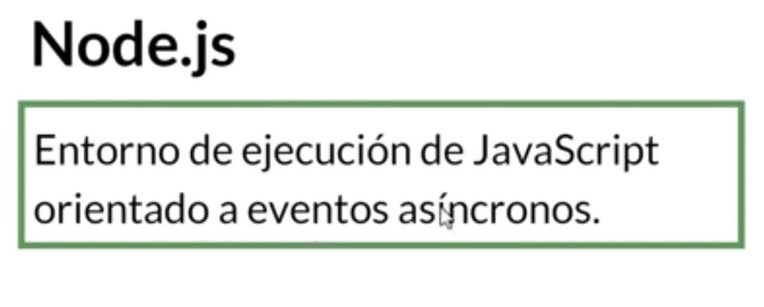
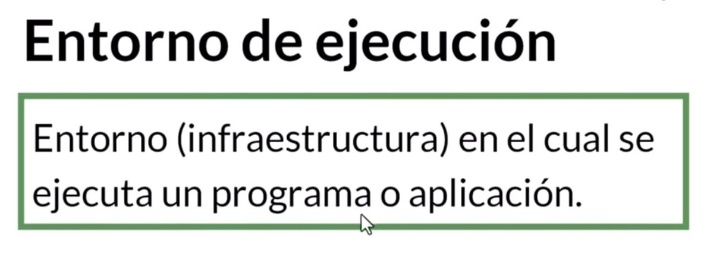
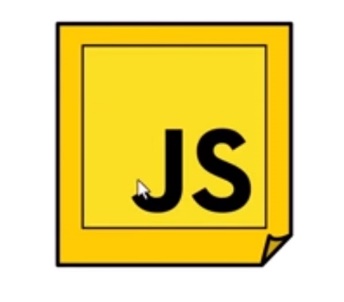
Curso de Node.Js Express y fundamentos de desarrollo WEB Backend

¿Qué es Node.js?

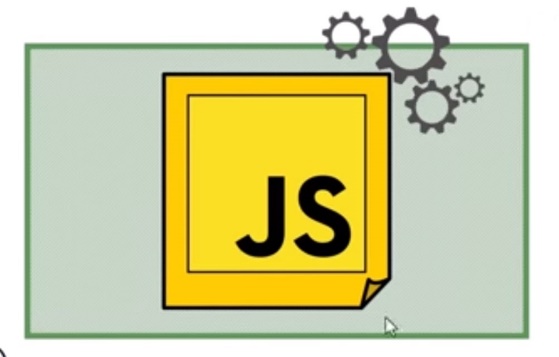


Vamos a desglosar cada una de las partes de este concepto porque tienen bastantes términos que probablemente te resulten nuevos. ¿Qué es un entorno de ejecución?

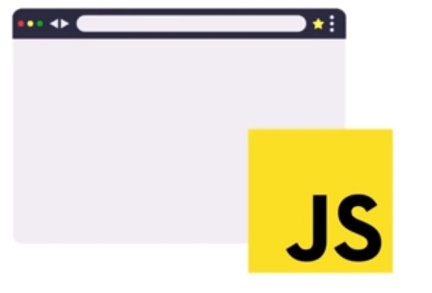




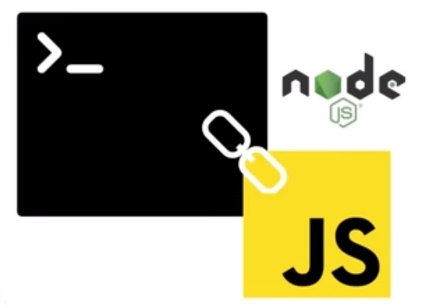
Si tenemos un archivo de JavaScript y queremos ejecutarlo todos eso componentes de software nos van a permitir ejecutarlo conforme a ese entorno de ejecución y Node nos va a dar todas esa herramientas



Anterior mente solo podíamos ejecutar JS en un Browser como googleChrone, internet Esplore, Mozila o cualquier otro tipo de navegador

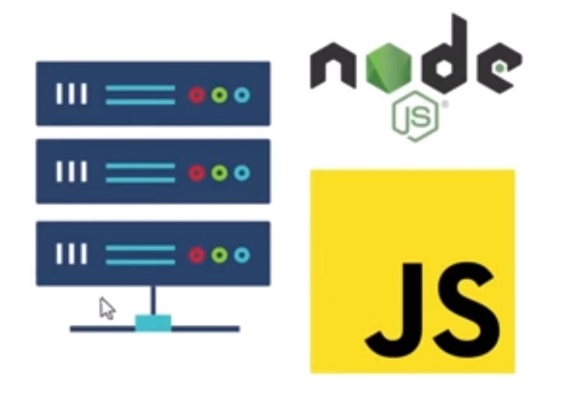


Pero ahora gracias Node podemos ejecutar código de JavaScript en el terminal.



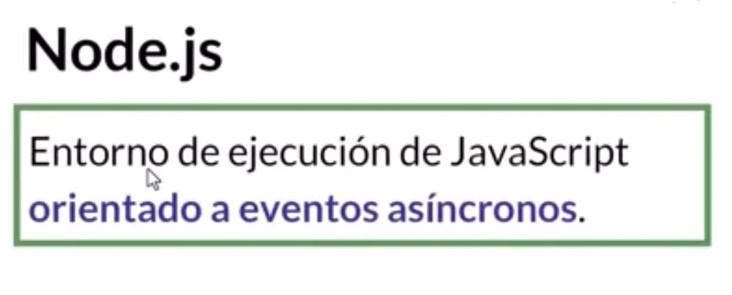
Sin necesidad de tener un navegador, simplemente podemos ejecutar nuestro programa directamente en la línea de comando de nuestro sistema operativo.

Vamos a usar estas herramientas que nos da Node para poder usar servidores y por eso es que NODE se

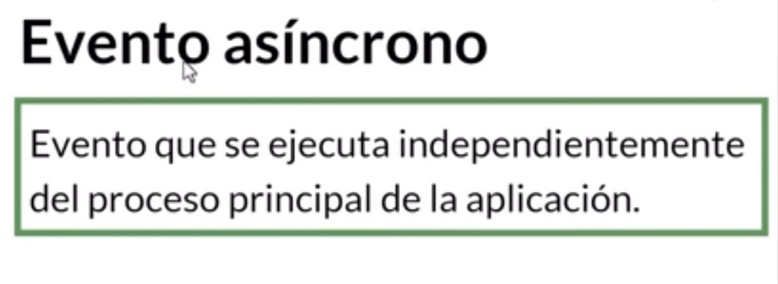


Usa principalmente para desarrollo Backend, desarrollos de los servidores que permiten que funcionen las aplicaciones Web que vemos y usamos todos los días.

Ya vimos que es un entorno de ejecución.



Y está orientada a eventos asíncronos, ¿Qué significa esto, que es un evento asíncrono?



Siguiendo este ejemplo:



Si tenemos una aplicación y esa aplicación se está ejecutándose en algún momento mientras estas ejecutando la aplicación quizás realices una acción, esa acción la vamos a considerar como un evento que desencadene un proceso.



Si ese proceso es asíncrono todo ese proceso que se desencadene va a continuar ejecutándose a medida que se ejecute la aplicación. Inicia luego ambos continúan su ejecución y el evento eventualmente se culmina y se continúa con la ejecución, no interrumpe de ninguna forma la ejecucion de la aplicación principal como por ejemplo cuando pedimos información a una página web o creamos un nuevo usuario en una plataforma eso toma un cierto tiempo pero no vemos que la aplicación se conjele o que ya no la podemos usar hasta que se complete el proceso y ese es el principio básico de un evento asíncrono.

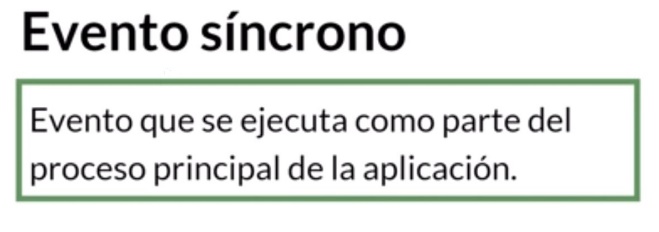


Por ejemplo:



Cuando necesitamos obtener información de una base de datos digamos que acabas de introducir tu usuario y contra seña esa acción de solicitar la información va desencadenar un evento asíncrono vamos a ir a un servidor luego a una base de datos y esa información tiene que ser retornada al Browser para que podamos verla pero ese proceso toma un tiempo unos cuantos milisegundos dependiendo de la red perola aplicación no se va a detener o congelar.

En contraste:



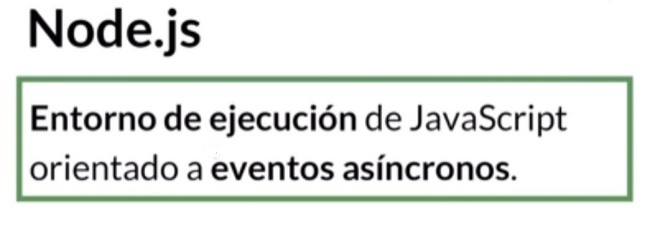
Los eventos asíncronos y continuaban su ejecucion en paralelo ahora con los evento síncronos si vamos a bloquear esa aplicación.



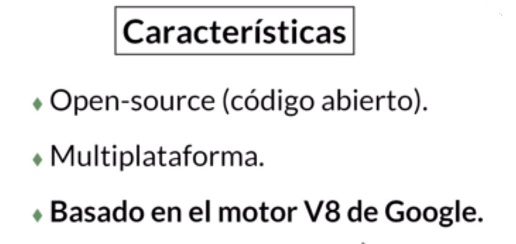
Si la aplicación se está ejecutando, cada evento síncrono que ocurra tiene que completarse antes de que la aplicación pueda continuar su ejecucion. Están como en la misma línea del tiempo no están ejecutándose en paralelo si no que es uno o el otro, mientra se ejecuta ese evento la aplicación va a estar bloqueada o congelada esa es la diferencia entre el evento asincrono y el evento síncrono.



Asique ya sabemos cada uno de los componentes de esa descripción o esta definición.



Aquí tenemos algunas características importantes de Node

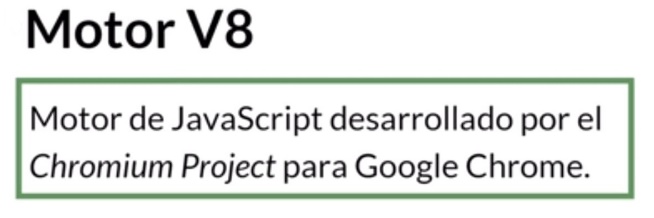


►Open-source podemos usarlos en nuestros proyectos en forma gratuita y también se puede contribuir a su código.

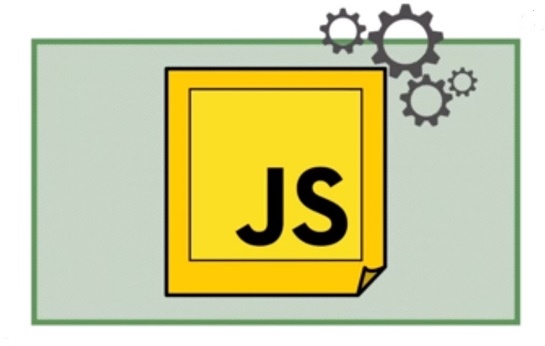
►Es multiplataforma es compatible con múltiples tipo de plataformas.

►Y está basado en el motor V8 de google básicamente es como un entorno donde nos permite ejecutar código en Google Chrome.

¿Qué es el motor V8?



Nos va a dar básicamente ese entorno de ejecucion que también nos da Node el terminal pero este motor nosnoda en el Browser en Google Chrome.



Node se basó en este entorno para poder ejecutar nuestro código en el terminal.

¿Qué es un motor de JavaScript?

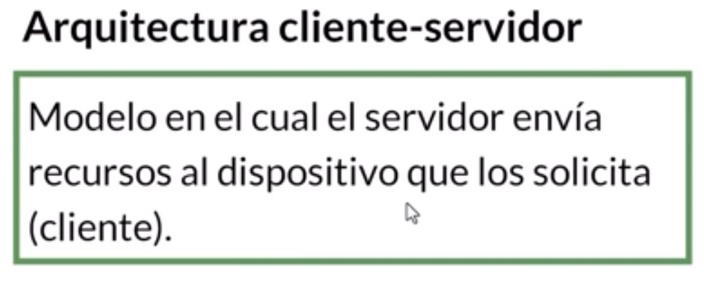


Bien, ahora veremos:



Que se deben conocer para trabajar con Node y desarrollo Backend.

Primero encontramos la arquitectura cliente servidor, también lo encontraran como modelo cliente servidor



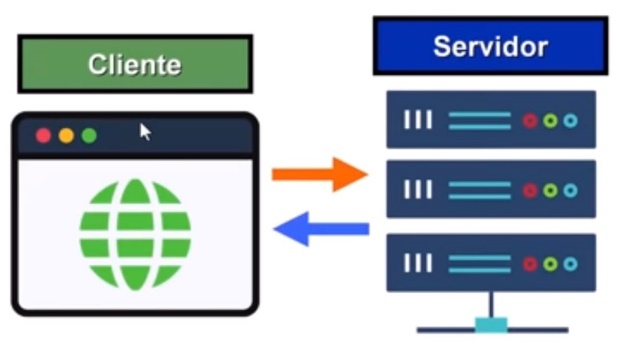
En la arquitectura tenemos un servidor y un cliente, por ejemplo:



Si estamos navegando en internet con nuestro Browser, digamos Google Chrome intentamos ingresar a esta página el navegador se encarga de ingresar esa solicitud.

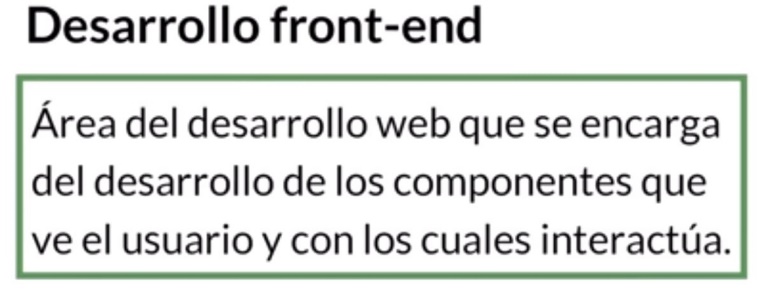


Y lego esa solicitud a travez de muchos pasos y de muchas redes llega a lo que conocemos como un servidor esa comunicación entre el cliente que solicita recursos o que solicita una acción y el servidor que está allí esperando escuchando esas solicitudes para poder responder se denomina el modelo o arquitectura cliente/servidor



Tenemos un cliente que solicita los recursos y un servidor que genera respuestas a partir de esas solicitudes.

Otro concepto importante para trabajar con Node es desarrollo Front-end y desarrollo Back-end son dos conceptos importantes.

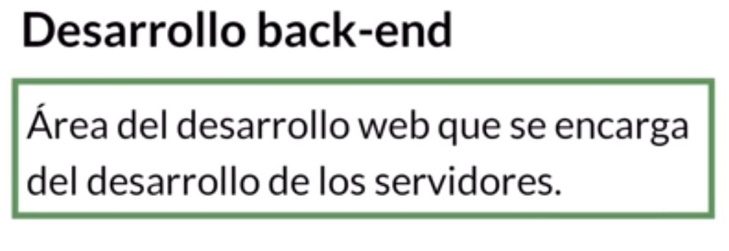


El desarrollador Front-end desarrolla todo lo que vemos como usuario



Con lo que interactuamos en el navegador la parte de la presentación y también la parte funcional con JavaScript, pero una vez que vamos un poco más allá en cuanto a información a poder guardar datos a poder

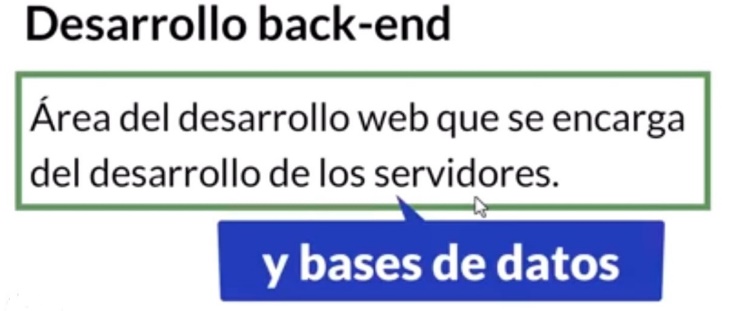
Registrar un usuario, entramos en otra área llamada:

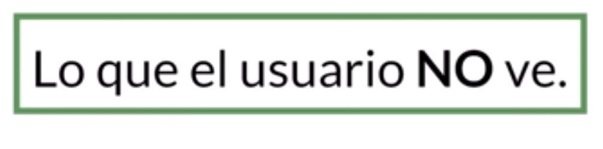


Recuerda que vimos el modelo cliente servidor básicamente el desarrollador Front-end desarrolla lo que vemos en el cliente.



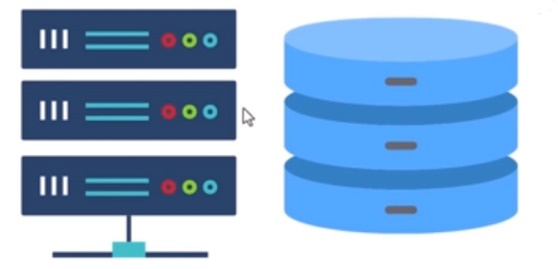
Y luego el desarrollador Back-end desarrolla lo que se va a ejecutar en el servidor para manejar esas solicitudes





El usuario no ve directamente lo que se implementa en un servidor pero ese servidor va a permitir que todo lo que se desarrolla para que si vea el usuario funcione correctamente.

Los servidores normalmente trabajan en conexión muy cercanas con bases de datos



Ellos son como unos intermediarios que van a manejar esas solicitudes para solicitar información de bases de datos o actualizar información en una base de datos.

Asique puedes considerar el desarrollo Front-end



Como si tuviéramos una tienda seria la parte frontal de la tienda todo lo que ve el cliente y el desarrollo

Back-end



Sería como el interior de la tienda, que permite que la tienda funcione pero no lo ve directamente el cliente Para eso usamos



Y en general si hablamos:

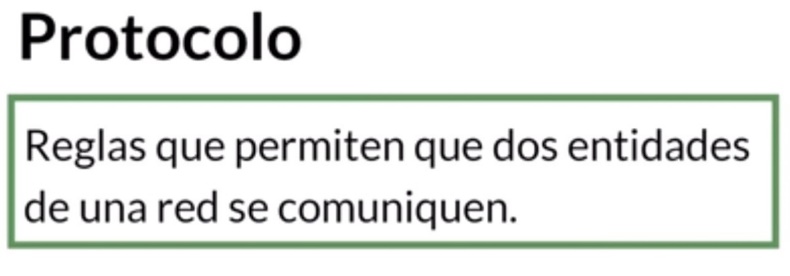


Combina ambas áreas el desarrollo Front-end y desarrollo Back-end para crear toda la aplicación que vemos

Ahora hablemos de otro concepto muy importante:



Cuando el navegador se comunica con el servidor no necesariamente van a estar en un formato que ambos entiendan como cuando intentamos hablar con alguien de otro país que no hable el mismo idioma, no entendemos ni nos entienden. Pero como se logró que funcione esta tecnología y esta comunicación, mediante el desarrollo de protocolos.



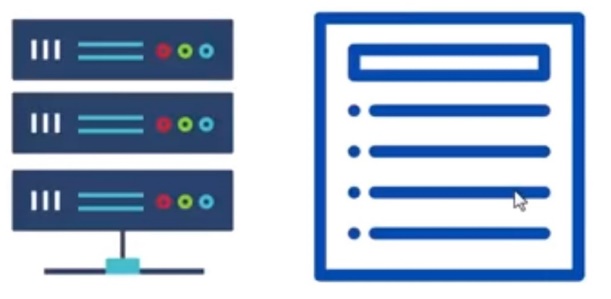
Cuando veas la palabra protocolo deberías pensar en la palabra *reglas.* Un protocolo es un conjunto de reglas que permite que se comuniquen, que el cliente entienda lo que dice el servidor y que el servidor entienda lo que dice el cliente.

El cliente:



El navegador va a enviar su solicitud en un formato específico.

Y el servidor también va mandar su respuesta en un formato específico



Y ambos van a estar coordinados



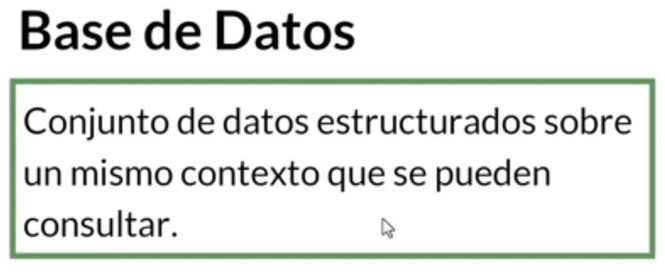
Para entender el formato del otro, el navegador va a entender lo que le envía el servidor y el servidor va a entender lo que le pide el navegador esa es la magia de desarrollar protocolo en la parte de redes.

En internet un protocolo muy importante y que vamos a ver en detalle es el protocolo



También es común que veas el protocolo **HTTPS** que es como una versión más segura y encriptada de HTTP Vamos hablar de http en detalle durante el curso. Eso en sintieses sería un protocolo.

Ahora hablemos de



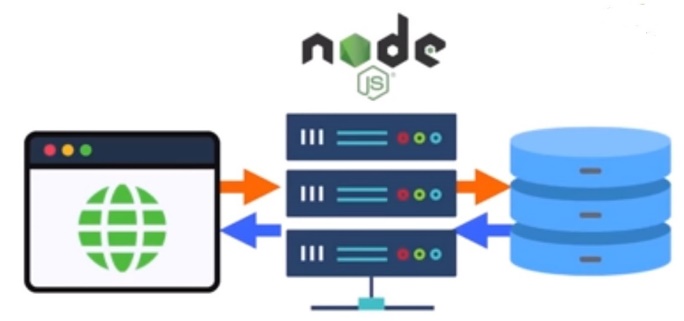
Vamos a tomar un conjunto de datos como por ejemplo los datos de nuestros usuarios, cada vez que te registras e ingresas un usuario y tu contra seña esa información se guarda en una base de datos son estructuradas por que se van a guardar de una forma específica en tablas dependiendo el tipo de bases de datos y normalmente son sobre un mismo contexto esos datos y esos datos se pueden consultar porque una base de datos no sería útil para nada si no pudiéramos solicitar esos datos y manipularlos.

Y esta es básicamente la estructura o la comunicación que resulta de este modelo



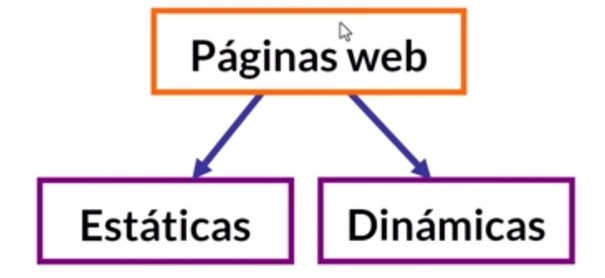
El navegador, el cliente se comunica con el servidor y luego el servidor se comunica con la base de datos luego la base de datos retorna la información o realiza la acción que se solicito y el servidor le manda una respuesta al cliente al navegador.

¿Y cuál es el rol de Node en este proceso?

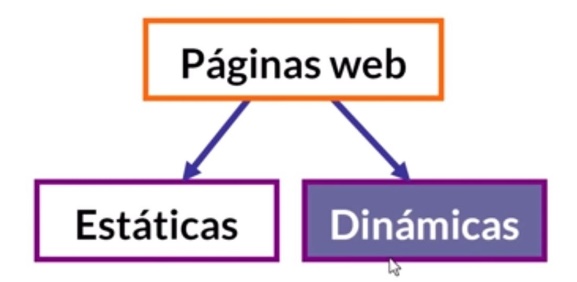


Bueno con Node vamos a desarrollar los servidores que van a procesar las solicitudes del navegador con Node.

Y finalmente dos conceptos importantes

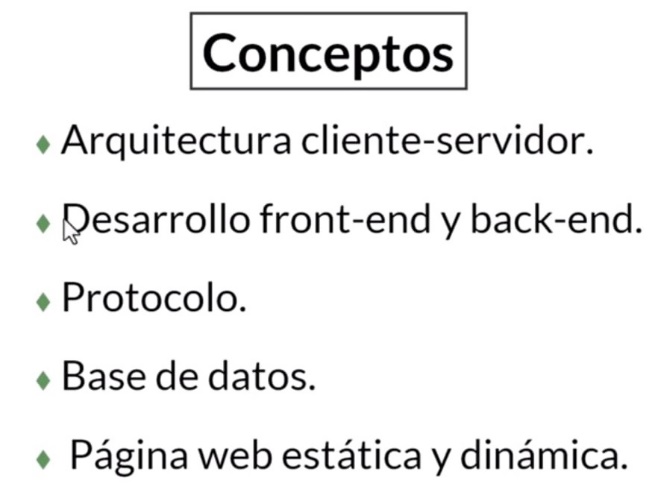


Son páginas web estáticas y páginas web dinámicas. Las páginas web estáticas son páginas una vez que están descargadas en el cliente, una vez que el navegador recibe todos los recursos. Los archivos, html, Css las imágenes ya no necesita ningún tipo de interacción con el usuario por ejemplo no se va a actualizar nada en la página, un ejemplo sería un block una vez que entras al artículo, este no va a cambiar permanecerá estático. A diferencia de las páginas web dinámicas.



Que si tienen interacción con el usuario y pueden realizar distintas solicitudes a los servidores.

Ya vimos conceptos muy importantes que vamos a usar en diferentes oportunidades en el desarrollo como un repaso qui vemos una lista.



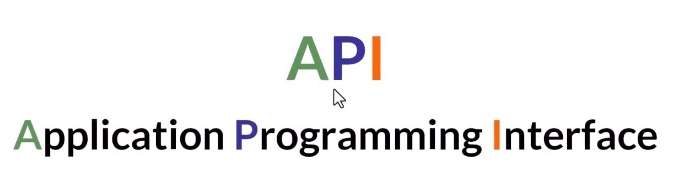
Para continuar ahora veremos:



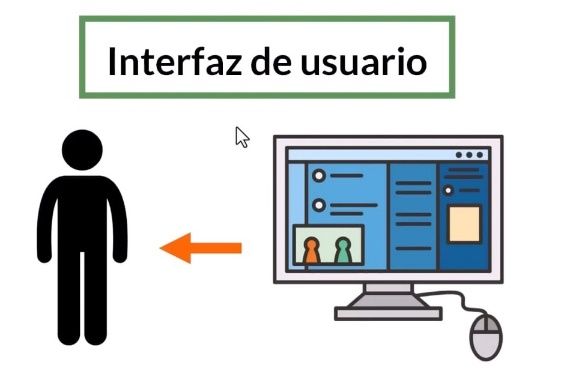
Ahora que ya conoces los conceptos básicos vamos a ver un poco más de detalles sobre las aplicaciones de Node, como lo podemos usar y que es lo que vamos a crear con Node.



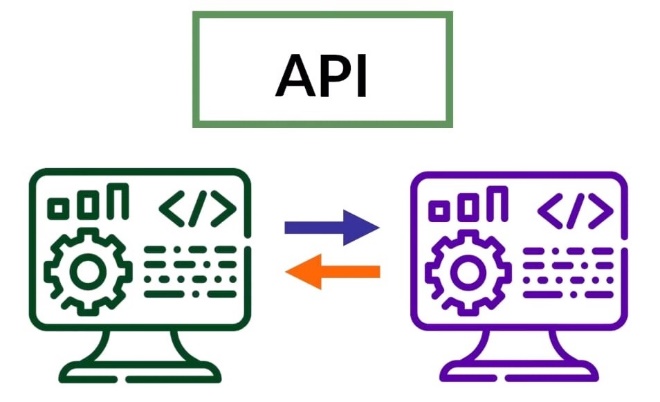
El desarrollo back-end es la parte de los servidores que interactúan con el navegador y con las bases de datos ahora veamos que son las APIS Una APIS es:



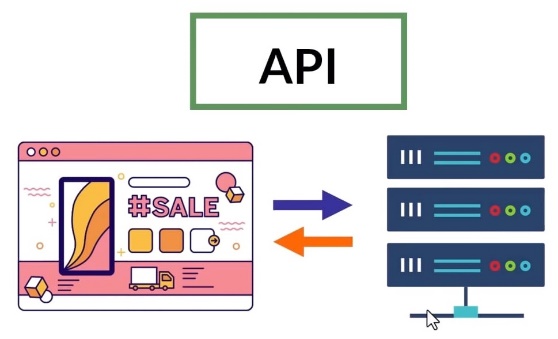
En español una interfaz de programación de aplicaciones es como un tipo de intermediario entre distintos programas. Cuando estas navegando en internet lo que ves.



Es la interfaz de usuario, esa interfaz de usuario actúa como una interfaz ara que el usuario pueda interactuar con esa página web. Todos esos elementos, los botones todo eso con lo que se interactúa se denomina la interfaz de usuario. Ahora con las APIS vamos a trabajar con otro tipo de interfaz.



Las API van a ser otro tipo de interfaz van a ser una interfaz entre distintos programas que se están ejecutando, en este caso el cliente y en el servidor.

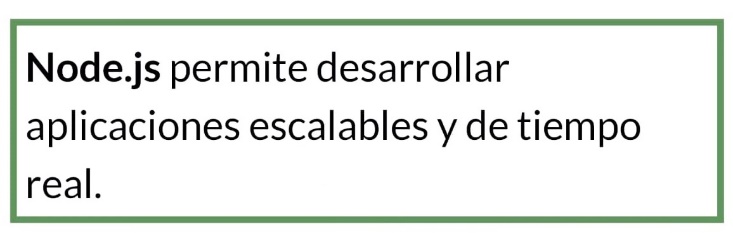


Esto va a permitirnos conectar la aplicación que se está ejecutando en el navegador con el servidor y así solicitar información a la base de datos o trabajar con ellas, también dentro del servidor podemos procesar solicitudes. No todo lo que vamos hacer en el servidor involucra una base de datos.

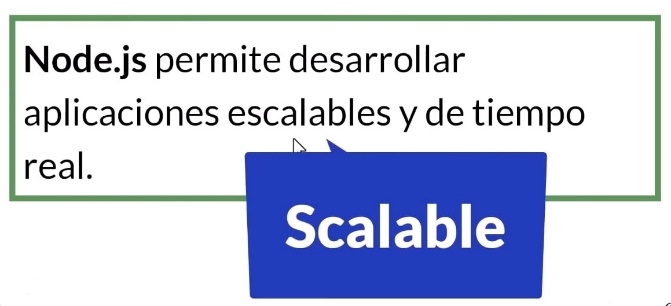
Cuáles son las ventajas de usar:



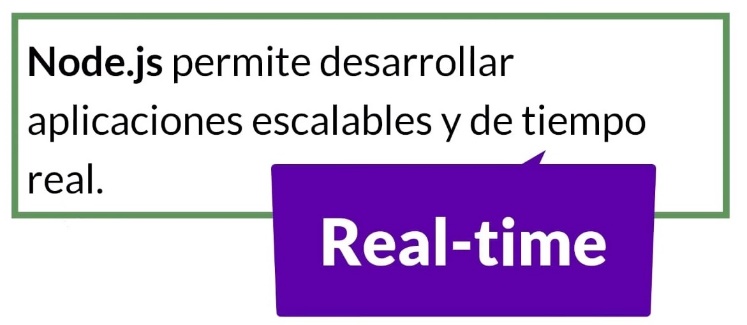
Sabemos que también existen muchas tecnologías alternativas para poder realizar servidores, Node nos permite desarrollar aplicaciones



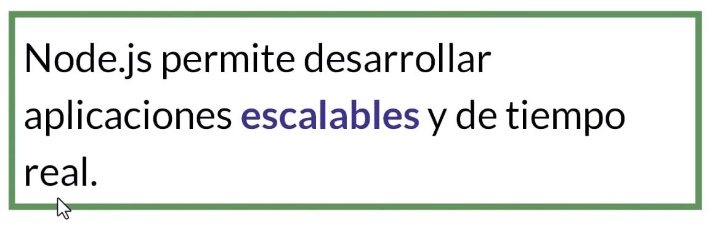
En ingles este término se pronuncia



Y tiempo real seria

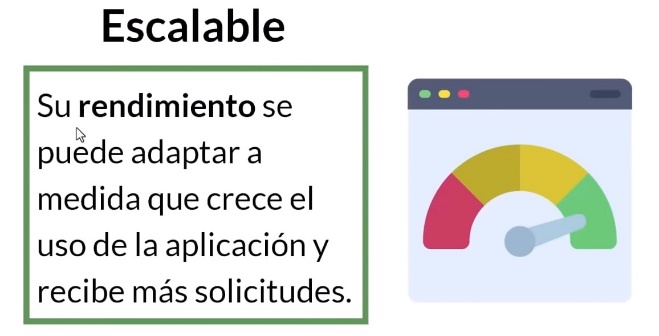


Y que significa escalable



Y que significa de tiempo real

Veamos: Una aplicación es

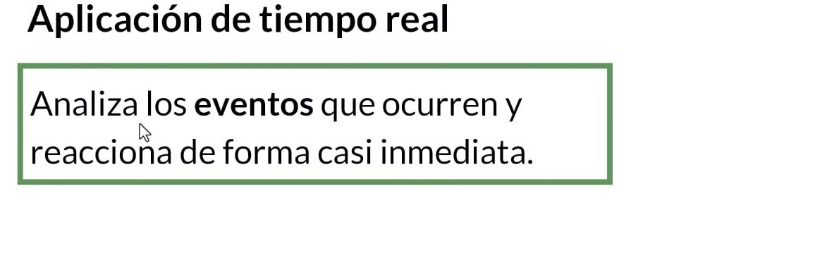


Por ejemplo si desarrollas una aplicación puede que funcione bien con los primeros 10, 20, 30 usuarios pero si en algún momento la aplicación incrementa ese número de usuarios de a miles y tienes que manejar esos números de usuarios, allí es donde se pone a prueba si tu aplicación es escalable o no, asique Node nos permite construir aplicaciones escalables es algo muy útil.

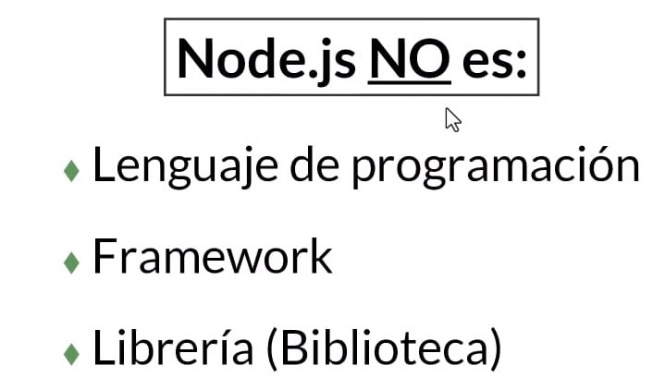
Y también nos permite construir aplicaciones



En esa conexión vamos a poder intercambiar información, el cliente puede solicitar información y el servidor le va a retornar esa información. Ese tipo de aplicación

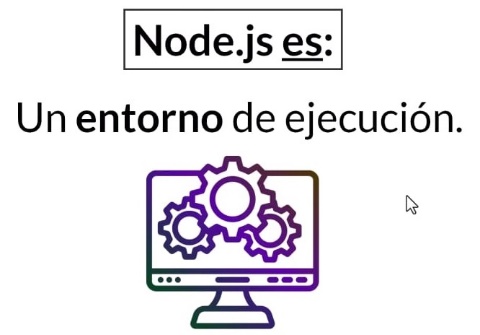


Que no es NODE:



Es importantes que sepas que Node **no es un lenguaje de programación**, Node es un entorno de ejecucion de JavaScript para escribir nuestros servidores. No es un **framework** y tampoco no es una **librería** no es ninguna de estas tres cosas.

Node es:



Nos va a dar todo ese entorno, todas esas herramientas para poder ejecutar nuestro código de JavaScript en el terminal. Dicho lo anterior



Una vez que hemos finalizado la instalación de NODE.JS vamos a verificar que la app se instaló correctamente en el terminal. **Verificar versión:** Ahora que se instaló Node lo primero que hay que hacer es comprobar que está instalado y que se agregó el comando Node al Path eso lo vamos hacer en la línea de comandos el terminal que tenemos en winwdos. Para abrirlo buscamos **command prompt** una vez abierta la aplicación en el prompt escribimos **node - -version** presionamos Enter y vemos nuestra versión v20.10.0 ya sabemos que si tenemos ese comando. Ahora vamos a experimentar con el REPL de Node que es como una herramienta interactiva donde podemos escribir código y ejecutarlo. Pero que significa REPL? RAED, EVAL, PRINT, LOOP en español seria el siclo de leer, evaluar y mostrar. Lo que vamos hacer es entrar en esa herramienta que nos va a permitir escribir nuestro código lo va a evaluar y nos va a mostrar el resultado de forma interactiva directamente en la línea de comando para ello escribimos el comando **node** nos da la bienvenida y versión solicitando que escribas HELP veamos que hace el comando .help cuando lo ejecutamos vemos algunos comandos *.break, .clear, .break, .editor, .exit, .help, .load, .save* cada uno tiene una función distinta, **.editor** te permite ingresar en el modo de editor, **.exit** te permite salir de esta herramienta de REPL, **.help** te permite mostrar este mensaje este mensaje de ayuda es el comando que ejecutamos, **.load** te permite cargar código desde un archivo a una sección interactiva de REPL Y **.save** nos permite guardar todos los comandos usados en esta sesión de REPL en un archivo. Seguidamente ingresamos en el modo editor y veamos que ocurre. No dice que estamos en modo editor y que debemos usar el atajo del teclado Ctrl + D para finalizar y Ctrl + C para cancelar. Este modo nos permite escribir código de JavaScript para evaluarlo directamente en el terminal. Digamos que queremos definir una función para mostrar Hola y el nombre de una persona como lo haríamos en VSC function saludar(nombre) { presionamos shift+enter para próxima línea y vamos a retornar una plantilla literal para poder remplazar ese parámetro `Hola $nombre`; serramos } y en la siguiente línea llamamos a la función con un nombre para ejecutar el código presionamos Ctrl+D y ahí vemos el resultado. Esto es básicamente el REPL de Node es una herramienta para experimentar y probar distintas funciones o distintos elementos de Node. Pudes salir del modo editor con Ctrl+C y para salir del REPL de esta herramienta interactiva porque mientra veamos el indicativo del prompt **>** vas a estar dentro de REPL para salir debes ejecutar el comando .exit y volvemos a ver lo que normalmente vemos en la línea de comandos.

**Primer programa con Node.js**

Vamos a VSC y creamos una carpeta cursoDeNode y creamos un archivo app.js donde vamos a escribir el mismo código que escribimos en REPL y asi entender por que